High Speed 64-kb Cache for Mobile Node

Robert Costanzo, Michael Recachinas, Hector Soto
ECE 4332 – Fall 2009
University of Virginia
<rwc3bf, mgr3yp, hls6dc>@virginia.edu

ABSTRACT

This paper presents methods for optimizing SRAM cell to build a high-speed 64kb SRAM cache. We detail our techniques yielding our design. Simulations are provided to illustrate our design choices. We used 45nm FreePDK technology for testing. Although we focus on optimizing the delay of our SRAM, we also optimized for power and area specifically focusing on the architectural and block levels. The overall goal of this paper is to prove to Portable Instruments Company (PICo) that our design is optimal. Our final metric was $5.29 \cdot 10^{-35}$ J · s^2 · mm^2 · W.

1. INTRODUCTION

Portable Instruments Company (PICo) requested a high speed 64kb SRAM for a mobile node with 32-bit words designed in the 45nm FreePDK. The metric we were asked to optimize in our design was *Active Energy Per Access · Delay*^{2} *· Area · Idle Power*, which places more emphasis on delay. The paper will detail the various layers of abstractions in elements used in the design of our SRAM, starting with micro-architecture then moving down to block-level, and finally ending with device level. This process of descending through layers of abstraction will also be used for the periphery used in the final integrated layout. Novelties in our design will be highlighted and justified. Finally, we quantify and discuss our final results and provide conclusions and comments on the design process.

2. SRAM ARRAY OVERVIEW

2.1 Introduction

Our SRAM design is divided into two types of components: global and local. Global components are those that affect the entire block of high-speed memory, while local components are block specific. The inclusion of local components allows for us to modify local knobs and view changes on a smaller scale prior to potentially implementing them elsewhere in the chip. Figure 1 shows the block diagram for our entire SRAM design.

2.2 Architecture Design

The first design knob that we turned was at the architecture level, specifically involving the number of blocks. After some research, we discovered a plot of number of divided blocks, n_B, versus normalized column power and normalized word-line delay (see Figure 2). We sought to remain on this Pareto frontier, specifically by using either 8 or 32 divided blocks. We chose 32 divided blocks because it seemed to provide very minimal column power usage (~0.02 normalized), while also maintaining a fairly small word-line delay metric (~0.35 normalized). We did not choose 8 blocks because we had other optimizations that focused on delay more. Furthermore, we decided that here was an ideal area to reduce power as much as possible.

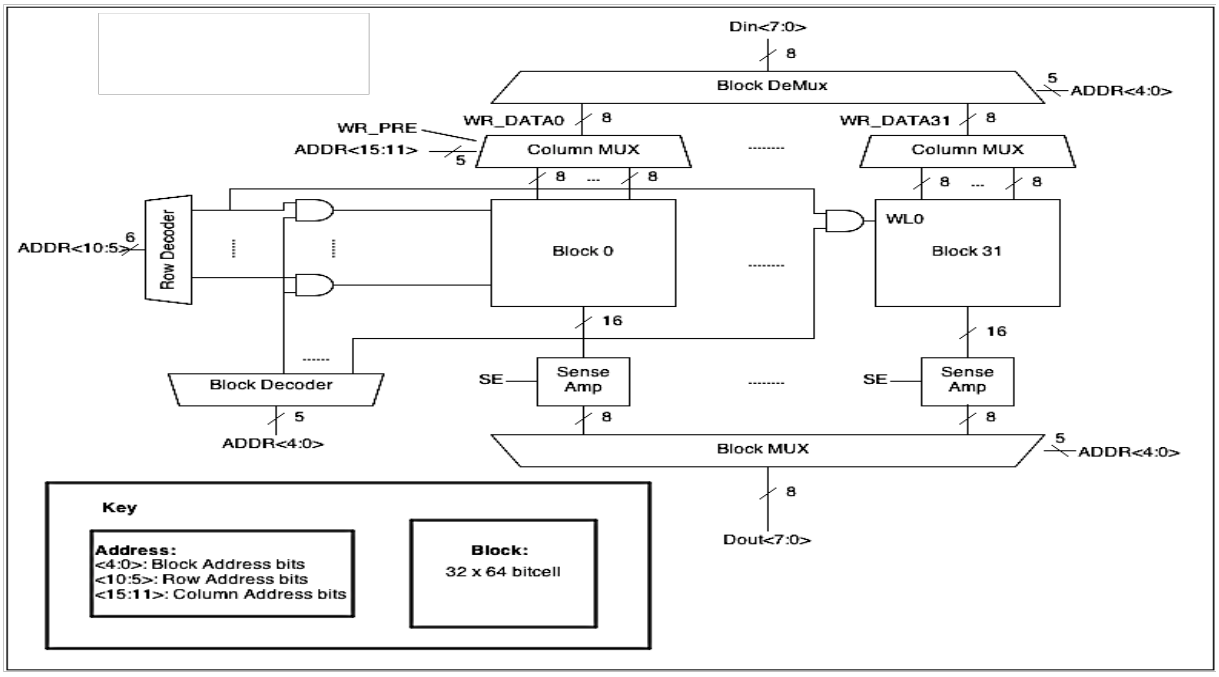

Figure 1. Global Block Diagram [1]

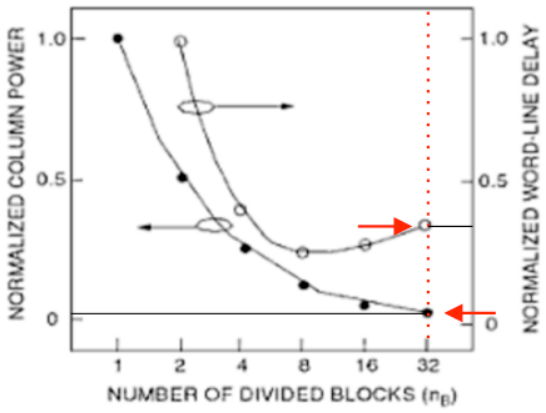

Figure 2. Pareto Curve for Divided Blocks [2]

3. BITCELL

3.1 Sizing

The logical next step after architectural optimizations involved block component and bit cell optimizations. At this level, logical effort sizing was the largest knob that we could turn. More specifically involving the bit cell, we were optimizing cell ratio and pull up ratio. These ratios are closely coupled with V_T, and vary for various device technologies. Before simulating with various ratios, we researched several measurements of ratios and the effect they had on other parameters. The following Figures 3 and 4 show several Pareto distributions involving ratios. We ultimately chose a cell ratio of 1.2 and pull up ratio of 1.11 [5] based on sweeping various ratios, predetermined by looking at previous groups' research. Ideally, we wanted to use minimum sized widths, but we realized this would increase delay, so we simulated larger widths. Once again, we ultimately agreed with Carr et al. that WPU = 180nm, WPD = 240nm, and WPG = 200n [5], which are verified by our simulations.

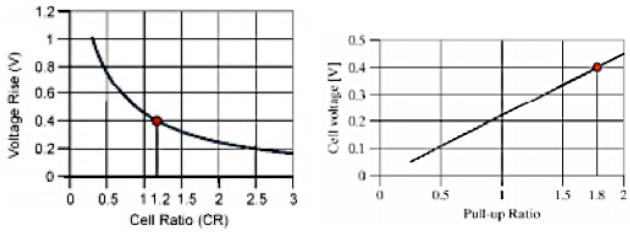

Figure 3. Cell Ratio versus Voltage and Stability [3]

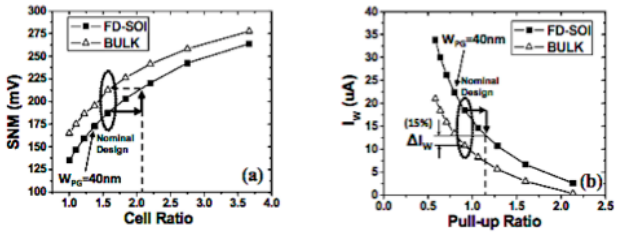

Figure 4. Cell Ratio and Pull-up Ratio versus SNM and I [4]

3.2 V_{DD}

As our group emphasized speed, we wanted to use the maximum voltage possible for our supply. Originally we utilized 2.5V for our supply, only to find out later that 45nm technology has a max V_{DD} of approximately 1.3V. Therefore, we utilized a maximum voltage of 1.1V, while leaving some space for a 10% error margin. In other words, we reduced the risk of our transistors being overdriven by accounting for a possible 10% increase in V_{DD} due to voltage supply variation.

4. PERIPHERY ANALYSIS

4.1 Decoders

There are two types of decoders used in the SRAM design: block and row. The decoders were originally implemented with transmission gates, but we discovered they did not drive the outputs as desired and are thus being implemented with traditional static CMOS logic. This way, the signals could achieve full rail-to-rail voltage swing, which was missing from the transmission gate implementation. The block decoder takes 5 bits from the given address and determines which block will be accessed. The row decoder takes a separate 6 bits from the address to determine which row from the specified block will be accessed. The output of the block decoder is ANDed with the output of the row decoder to ensure only the row in the particular block is accessed, further lowering energy consumed with minimal impact on speed.

4.2 Block MUX/DeMUX

The MUX's are used to handle data propagation during the read. The block DeMUX utilizes 5 bits from the address signal and sends the input data to the appropriate block. The column MUX then utilizes another 5 bits from the address to select which column (BL/BLB set) within the block the data is sent to for a write. During a read, the block output MUX selects which sense amp to read and pass to the output. All of the MUX's were built with a transmission gate design as we ascertained that floating inputs and outputs did not cause issues between reads and writes. However, there are buffers within the designs to produce full swing signals.

4.3 Pre-charge

Pre-charging of the bit line (BL) and bit line bar (BLB) requires two PMOSs to be connected to VDD via their sources and their drains to BL and BLB. When the lines need to be pre-charged the PMOSs are turned on, allowing BL and BLB to be connected to VDD. The PMOSs used in charging up the bit lines were upsized significantly to speed up the charging process. Not only that, but in order to reduce the differential in the BL and BLB that may impact a read, a third PMOS was added, connecting BL and BLB during the pre-charge stage.

At the other end of the BL and BLB are NMOS devices. These NMOS devices act as the write drivers and connect BL and BLB to Data and DataB (respectively) when the write signal is high. These devices also had to be upsized to ensure faster and more reliable data writes.

4.4 Word Lines

The word line signals are selected by the process described in section 4.1. The two decoders together produce local word line signals that ensure only the necessary line is driven, and thus only the necessary row is accessed. This process does two things: only one row decoder is needed for all thirty-two blocks, and it reduces the overall power as only the needed word-line is charged when it is required.

4.5 Sense Amplifier

The sense amplifier is an analog circuit that receives both the bit line and bit line bar and senses a difference between the two. If bit line is greater than bit line bar, the output will be high; otherwise, the output will be low. Several common topologies exist: differential voltage sense amplifier, current mirror sense amplifier, and latching voltage sense amplifiers to name a few. Knowing that the current mode sense amplifier could sense at smaller deltas between the bit lines, we initially planned to use this design; however, given time constraints, we chose the latching voltage sense amplifier topology from [6] due to a simpler yet still relatively optimal implementation. In the final design, however, we ran into problems with the amplifier's output. Therefore, we reverted to the most common and simplest implementation—the differential voltage sense amplifier with buffered output. The consequences of this decision were 23% increase in read delays [7], but smaller area overall (differential sense amplifier - 9T versus latching voltage - 13T).

5. SIMULATION AND RESULTS

5.1 Test Bench

In order to verify our design's functionality, we opted to create a representative test bench as opposed to a full simulation. As such, we set up a 2x2 bit cell array. However, to maintain more accurate operating conditions, we loaded the BL and BLBs with 62 instances of bit cells. We loaded the WL's with 30 instances of bit cells. This would recreate parasitics similar to the full array. Also, we loaded the wires with additional capacitances representing wire capacitances extracted from layout. The decoders and mux's were smaller variants of the full versions, taking in only 1 bit of address per decoder as opposed to the full 5. With this setup, we could test the full arrays delay and functionality while focusing only on 4 bit cells (within the 2x2).

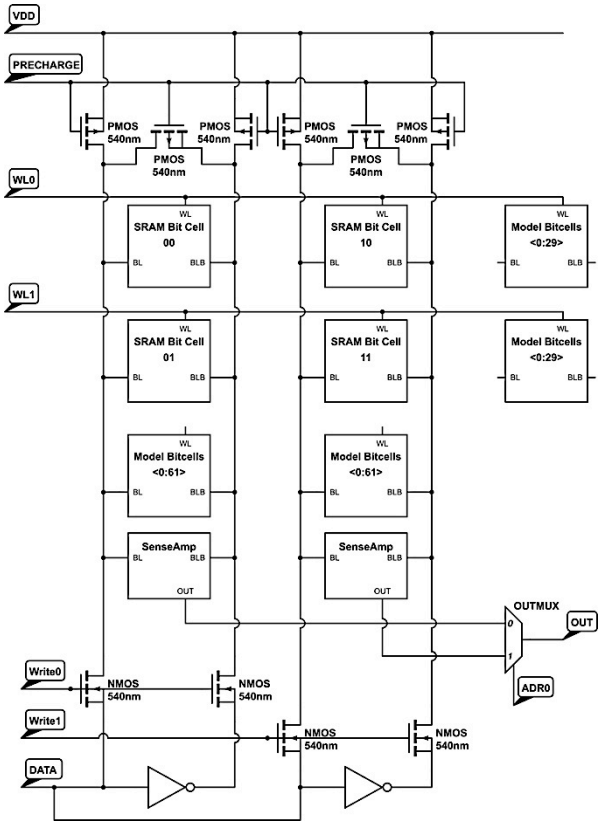

Figure 5. 2x2 Block Model Test Bench to Simulate SRAM

5.2 Results

Ultimately, the simulations from our representative model proved promising. We were able to write and read successfully from bit cells in different columns and rows. This not only verified the functionality of our SRAM bit cells, but also the decoding process. The signals within the bit cells (Q and QB) tend to glitch at times, but the magnitudes of the glitches never surpassed 200mV so they are not very concerning.

The simulation had a simple test algorithm. We would first write a one, a zero, and then another one to Q10; between each write, we'd read to make sure the proper value was stored in Q and that we were able to read zeroes and ones. Then, we switched to bit cell 01 and wrote a zero, a one, and then another zero, also while reading in between writes. The contents of the bit cells are shown below. Out of interest of space, we have neglected to show the inputs, bit lines, and decoder outputs, and instead do our best to label the outputs shown. However, we see full voltage swing signals and everything appears to function ideally.

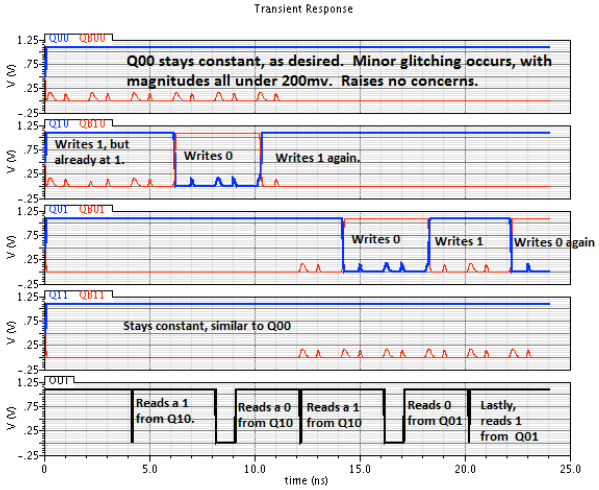

Figure 6. Stored Values of 2x2 SRAM Test Bench

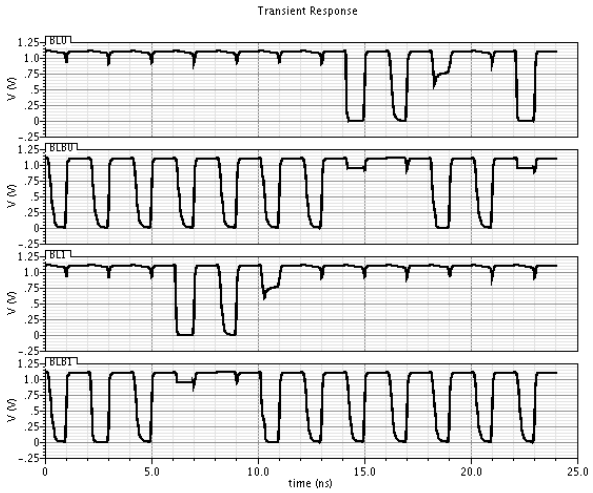

Figure 7. Plot of BL and BLB corresponding to Figure 4

5.3 Metrics

Table 1. Metrics for High Speed Cache

Total Metric	$5.29 \cdot 10^{-35}$ J · s^2 · mm^2 · W
1 Bit cell Area	2.3 μm^2
Total Area	0.1657 mm^2
Total Energy	16.7640 pJ
Read Delay	0.1392 ns
Write Delay	0.2501 ns
Total Delay (WC)	0.2501 ns
Idle Power	0.3047 mW
Maximum Reliable Frequency	0.5 GHz

To obtain the area metric, we found the area of a block of SRAM and a Decoder and multiplied by 32 to approximate for 32 blocks. The read delay was simply the time for the output to reach $0.5 \cdot V_{DD}$ after the read signal had reached $0.5 \cdot V_{DD}$. Similarly, the write delay was time necessary for Q within a cell to reach $0.5 \cdot V_{DD}$ in a transition after the write signal had reached $0.5 \cdot V_{DD}$. Idle power was calculated by leaving all inputs static, other than the clock (pre-charge), finding the current out of V_{DD} and then multiplying by V_{DD}. The total energy was found using $I_{\text{average from VDD}} \cdot V_{DD}$ · Access Time. The final metric was found by multiplying all of these while squaring the delay as we emphasized speed. We stepped up frequency and found our model failed to work as desired at 1GHz, so we use 0.5GHz.

An interesting observation to note: In general, write speeds are faster than reads. However, we found the opposite. This is because the decoders that influence the speed of the writes were implemented with static CMOS gates, and the MUX that influences the read speed was made with transmission gates, the read speed was actually faster. This allowed us to use the write speed as our critical delay as opposed to a slower read delay.

6. CONCLUSION

In conclusion, we believe our group has a functional, early model for PICo's design specifications for a high-speed cache on a mobile node. In order to maximize speed, we used the maximum allowable V_{DD} of 1.1V. While other options were available, the group decided to utilize the standard 6T cell that we discussed in class. It was also discovered that while transmission gates are fast and low area, they were unable to drive full-swing signals and did not make for quality decoders. While we were able to recreate a static CMOS implementation of a decoder for the model, we had no such luck in the full 32-output decoder due to time constraints. Given more time, we would also like to implement pre-decoding in our new decoders to create faster decoding of address bits and error-correcting-code (ECC) to increase the robustness and integrity of our design. Ultimately, we are satisfied with the new model and decoders' ability to show that our SRAM functions as desired.

Throughout the entire design process, trade offs remained quite prevalent. Every modification we made to the design required even more changes for the SRAM to work appropriately. We sacrificed some energy efficiency by driving the voltage supply to maximum, but as our metric emphasized speed, we believe it was justified. If allowed to work on the design further, we may consider implementing a SLEEP mode in the device as we do believe that the added energy saved would be worthwhile. Furthermore, we would replace the differential sense amplifier with the current mirror sense amplifier to decrease read delay. Lastly, at the architecture level, we would move further along the Pareto curve in Figure 2 and test 8 blocks, thereby optimizing both delay and power consumption together.

7. ACKNOWLEDGMENTS

We would like to thank Professor Benton Calhoun, Aatmesh Shrivastava, and Divya Akella for their guidance and support throughout this project. We would also like to thank all of the previous project groups for their advice and hints regarding designing an optimal SRAM cell. Finally, we would like to thank the University of Virginia, with special emphasis placed on the Charles L. Brown Department of Electrical and Computer Engineering, for providing the resources for the project.

8. REFERENCES

[1] Cash, L. E., Duan, C., Reed, R. C., Tyler, A. D. (2012). Pico Embedded High Speed Cache Design Project. University of Virginia. Charlottesville, VA.

[2] Pearson, A., DeNardi, S., Vaughan, E., and Boley, J. (2009). A High-Speed SRAM Cache in AMI 0.6μm Technology. University of Virginia. Charlottesville, VA.

[3] Rabaey, J. M., Chandrakasan, A. P., & Nikolic, B. (2002). *Digital integrated circuits* (Vol. 2). Englewood Cliffs: Prentice hall.

[4] Shin, C. (2011). Advanced MOSFET Designs and Implications for SRAM Scaling (30). University of California, Berkeley. Berkeley, California.

[5] Carr, D. A., Park, J. P., Reyno, D. A. (2010). A High Speed 64kb SRAM Cache in 45nm Technology. University of Virginia. Charlottesville, VA.

[6] Ryan, J. F., & Calhoun, B. H. (2008, March). Minimizing offset for latching voltage-mode sense amplifiers for sub-threshold operation.

[7] Chen, Y., Converse, C. R., Gan, C., Moore, D. M. (2010) Team 2 Implementation of a Low Power SRAM design using 45 nm FreePDK Technology. University of Virginia. Charlottesville, VA.